CLAIMS

What is claimed is:

1. A method for providing exactly-once semantics for web-based transaction processing in a system having a client and a server, comprising the steps of:

a) the client requesting a form from the server;

b) the server generating a unique identifier for identifying a particular transaction;

c) the server providing a form with the unique identifier to the client;

d) posting a filled out form to the server; wherein the filled out form includes the unique identifier;

e) upon receiving the filled out form, the server generating a status page for informing the user that the transaction is being processed and returning the status page to the client; and

f) after returning the status page, the server performing transaction processing.

2. The method of claim 1 wherein the step of after returning the status page, the server performing transaction processing includes

checking to determine if there is a transaction with the current unique identifier that has already been committed.

3. The method of claim 2 wherein the step of checking to determine if there is a transaction with the current unique identifier that has already been committed includes the steps of

performing a rollback operation on in-progress transactions with the same unique identifier; and

determining the result and outcome of transactions that have the same unique identifier.

4. The method of claim 3 wherein when the outcome is abort, executing a new transaction with the same form data.

5. The method of claim 3 wherein when the outcome is commit, providing the result to the client.

6. The method of claim 5 wherein the step of when the outcome is commit, providing the result to the client includes the steps of

providing the transaction outcome to a testable transaction abstraction; and

providing the transaction result to the testable transaction abstraction;

wherein the transaction outcome and the transaction result of the transaction are highly available.

7. The method of claim 1 further comprising the step of:

storing the unique identifier in a uniform resource locator (URL).

8. The method of claim 1 further comprising the step of:

storing the unique identifier in a browser cookie.

9. The method of claim 1 wherein the form provided to the client in step c) includes at least one instruction that instructs the user to reload the current page when a failure message is displayed.

10. The method of claim 1 wherein posting the filled out form to the server includes the step of a user filling out the form.

11. The method of claim 1 further comprising the step of

automatically reloading the status page after a predetermined time interval; wherein the client can automatically check the status of a transaction without user involvement.

12. The method of claim 1 wherein the status page includes the unique identifier and form data.

13. The method of claim 1 wherein the step of after returning the status page, the server performs transaction processing includes

executing server-side business logic.

14. The method of claim 1 wherein the client includes a web browser.

15. A web-transaction processing system for providing exactly-once semantics for web-based transaction processing comprising:

a) a client for requesting a form from the server and allowing a user to fill out and submit forms; and

b) at least one server for generating a unique identifier for identifying a particular transaction; providing a form with the unique identifier to the client; upon receiving the filled out form, the server generating a status page for informing the user that the transaction is being processed and returning the status page to the client; and after returning the status page, the server performing transaction processing.

16. The web-transaction processing system of claim 15 further comprising:

a) an automatic retry mechanism for automatically retrying transactions whose previous transaction attempts have an outcome of abort without user intervention; wherein the automatic retry mechanism employs a unique identifier for identifying a particular transaction.

17. The web-transaction processing system of claim 16 wherein the automatic retry mechanism performs a retry at a predetermined time interval until the outcome is a commit.

18. The web-transaction processing system of claim 15 further comprising:

a mechanism for enabling a user to safely retry a transaction that previously failed.

19. The web-transaction processing system of claim 15 further comprising:

a status checking module for prior to re-executing server-side business logic, checking the status of previously executed transaction with the same transaction identifier.

20. The web-transaction processing system of claim 15 further comprising:

a status update module for when the outcome is commit, providing the transaction outcome to a testable transaction abstraction, providing the transaction result to the testable transaction abstraction, and providing the result to the client;

wherein the outcome and the result of the transaction is highly available.

21. A method for handling errors in applications that employ a request-reply protocol in a system having a client and a server, comprising the steps of:

a) the client requesting a form from the server;

b) the server generating a unique identifier for identifying a particular transaction and providing the form to the client; wherein the form includes the unique identifier;

c) the client filling out the form and providing the filled out form to the server;

d) the server generating a status page to the client;

e) when the client has received a status page, the client automatically checking the status of the transaction, when the transaction has failed, the client retrying the transactions after a predetermined time interval without user involvement providing exactly-once semantics;

f) when the client has not received the status page, involving the user in error handling by providing at-most once semantics.

22. The method of claim 21 wherein the step of when the client has not received the status page, involving the user in error handling by providing at-most once semantics includes:

instructing a user to follow a predetermined link to a status page.

23. The method of claim 21 wherein the client-side retry mechanism is implemented by utilizing HTML and HTTP semantics; and

wherein the client-side error handling mechanism is embedded in downloaded pages.